A Declarative Framework for Security: Secure Concurrent Constraint Programming

Hugo A. López[1], Catuscia Palamidessi[3], Jorge A. Pérez[1], Camilo Rueda[1], and Frank D. Valencia[2]

[1] Pontificia Universidad Javeriana Cali
{halopez,japerez,crueda}@cic.puj.edu.co
[2] INRIA and LIX École Polytechnique catuscia@lix.polytechnique.fr
[3] CNRS and LIX École Polytechnique frank.valencia@lix.polytechnique.fr

Motivation. Due to technological advances such as the Internet and mobile computing, *Security* has become a serious challenge involving several disciplines of Computer Science. In recent years, there has been a growing interest in the analysis of *security protocols* and one promising approach is the development of formalisms that model communicating processes, in particular *Process Calculi*. The results are so far encouraging although most remains to be done.

Concurrent Constraint Programming (CCP) is a well-established formalism which generalizes *Logic Programming* [Sar93]. In CCP processes interact with each other by telling and asking information represented as *constraints* in a medium, a so-called *store.* One of the most appealing and distinct features of CCP is that it combines the traditional *operational* view of processes calculi with a *declarative* one of processes based upon logic. This combination allows CCP to benefit from the large body of techniques of both process calculi and logic. Over the last decade, several reasoning techniques and implementations for CCP have been developed: E.g., denotational models [SRP91], specification logics and proof systems [NPV02], Petri Net interpretations [RM94], and *CCP-based programming languages* [Smo95].

Remarkably, most process calculi for security have strong similarities with CCP. For instance, SPL [CW01], the Spi calculus variants in [ALV03,FA01], and the calculus in [BB02] are all operationally defined in terms of configurations containing information which can only increase during evolution. Such a monotonic evolution of information is akin to the notion of *monotonic store*, which is central to CCP and a source of its simplicity. Also, the calculi in [ALV03,BB02,FA01] are parametric in the underlying logic much like CCP is parametric an underlying *constraint system*. Also, the assertion of (protocol) properties [ALV03] can be formalized as CCP processes imposing constraints. Furthermore, the *notion of unification*, which has been shown useful in [FA01] for the symbolic execution of protocols, is primitive (and more general) in CCP.

Description. Our project *Secure CCP (SCCP)* aims at advancing both the theory and tools of CCP for analyzing and programming security protocols. The main goal is to develop a CCP-based framework for security protocols. The novelty is the combination in one unique formalism of behavioral and logical techniques. In fact, to our best knowledge, there is no work on Security that takes advantage of the reasoning techniques of CCP such as its denotational models, temporal and intuitionistic logics, or Petri Net

interpretations. The expected outcome is two-fold. We will advance the CCP theory to deal with new challenging concepts from Security and produce a specification language and tools to model and automatically verify security protocols.

Approach. The approach of the project will be to give a CCP account of a representative calculus for security protocols. We will use CCP *constraint systems* to represent a logic to reason about the information an attacker can deduce from the information accumulated in the monotonic store. The CCP linear-time temporal logic and associated complete inference system in [NPV02] and the verification results in [Val05] can be used to specify and prove safety properties of protocol runs.

Now, most security protocols use mechanisms to allow generation of nonces (or names). Therefore, we shall need to provide CCP with such mechanisms which have been far too little considered in CCP. One approach to this problem will be to use constraint systems based on Nominal Logic [Pit01], a modern logic to reason about *name freshness*. Another possibility is to extend CCP with an operation that provides name generation. To keep the dual operational and declarative view of CCP, the extended language should also have a logic interpretation. In fact, we have recently studied the issue of name generation in [PSVV06] where we proved that existential quantification can replace name generation in a meaningful process calculus.

References

[ALV03] R. Amadio, D. Lugiez, and V. Vanackere. On the symbolic reduction of processes with cryptographic functions. *TCS: Theoretical Computer Science*, 290, 2003.

[BB02] M. Boreale and M. Buscemi. A framework for the analysis of security protocols. *Lecture Notes in Computer Science*, 2421, 2002.

[CW01] F. Crazzolara and G. Winskel. Events in security protocols. In Pierangela Samarati, editor, *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pages 96–105, Philadelphia, PA, USA, November 2001. ACM Press.

[FA01] M. Fiore and M. Abadi. Computing symbolic models for verifying cryptographic protocols. In *14th IEEE Computer Security Foundations Workshop*, pages 160–173. IEEE Computer Society, 2001.

[NPV02] M. Nielsen, C. Palamidessi, and F. Valencia. Temporal concurrent constraint programming: Denotation, logic and applications. *Nordic Journal of Computing*, 9(2):145–188, 2002.

[Pit01] A. Pitts. Nominal logic: A first order theory of names and binding. In *Proc. of TACS 2001*, volume 2215 of *LNCS*. Springer-Verlag, 2001.

[PSVV06] C. Palamidessi, V. Saraswat, B. Victor, and F. Valencia. On the expressiveness of recursion vs replication in the asynchronous pi-calculus. To Appear in LICS'06, 2006.

[RM94] F. Rossi and U. Montanari. Concurrent semantics for concurrent constraint programming. In *Constraint Programming: Proc. 1993 NATO ASI*, pages 181–220, 1994.

[Sar93] V. Saraswat. *Concurrent Constraint Programming*. The MIT Press, 1993.

[Smo95] G. Smolka. The Oz programming model. In Jan van Leeuwen, editor, *Computer Science Today*, volume 1000 of *LNCS*, pages 324–343. Springer-Verlag, 1995.

[SRP91] V. Saraswat, M. Rinard, and P. Panangaden. The semantic foundations of concurrent constraint programming. In *POPL '91*, pages 333–352, 1991.

[Val05] F. Valencia. Decidability of infinite-state timed CCP processes and first-order LTL. *Theor. Comput. Sci.*, 330(3):577–607, 2005.